Queens College
Computer Science Department

CS 212 – Fall 2020 – Project 2

Assigned: 22 October 2020
Due: 2 November 2020
Cutoff: 7 November 2020

List of Words

Create a class called *WordNode* which has fields for the *data* (a Word) and *next* (WordNode) instance variables. Include a one-argument constructor which takes a Word as a parameter. (For hints, see the PowerPoint on "Static vs. Dynamic Structures".)

```
public WordNode (Word w) {  . . }
```

The instance variables should have protected access. There will not be any get and set methods for the two instance variables.

Create an abstract linked list class called *WordList.* This should be a linked list with head node as described in lecture. Modify it so that the data type in the nodes is *Word*. The no-argument constructor should create an empty list with *first* and *last* pointing to an empty head node, and *length* equal to zero. Include an append method in this class.

Create two more linked list classes that extend the abstract class *WordList*: One called *UnsortedWordList* and one called *SortedWordList*, each with appropriate no-argument constructors. Each of these classes should have a method called *add(Word)* that will add a new node to the list. In the case of the UnsortedWordList it will add it to the end of the list by calling the append method in the super class. In the case of the SortedWordList it will insert the node in the proper position to keep the list sorted.

Instantiate two linked lists, and for every Word read from the file, add it to the unsorted and sorted lists using the *add* method. You will end up with the first list having the Words from the input file in the order they were read, and in the second list the Words will be in sorted order. Display the unsorted and sorted Words in the GUI just as in project 1.

Submitting the Project.

You should now have the following files to submit for this project:

```
Project2.java
Word.java
WordGUI.java
WordNode.java
WordList.java
UnsortedWordList.java
SortedWordList.java
```

Submit a jar file.

Rather than upload all the files above separately, we will use Java's facility to create the equivalent of a zip file that is known as a **J**ava **AR**chive file, or "jar" file.

Instructions on how to create a jar file using Eclipse are on Blackboard. Create a jar file called **Project2.jar** and submit that. **Be sure the jar file contains *source* code**, not classes.